VIC-20
SCREEN MASTER

VIC-MASTER

User's Guide

and

Reference manual

COPYRIGHT

This software product is copyrighted and all rights reserved by MICRO APPLICATION.

Copying, duplicating, selling or otherwise distributing this product is a violation of the law.

This manuel is copyrighted and all rights are reserved.

WARNING

Although this software product has been tested thoroughly before release, MICRO APPLICATION and VIC-MASTER software distributors cannot assume any liability or responsability for any loss or damage arising for the use of this program. That point should be clearly understood when this software is bought.

Individual applications should be carefully tested before implementation and commercialization.

MICRO APPLICATION

73 rue de Turbigo
75003 PARIS
FRANCE

Tél : (1) 274.70.55
Telex : 270 105 F TXFRA/réf 311

Relase : MV02-ENG-09/82

1 Overview..... 1-1

2 Installation..... 2-1

3 Screen Generator..... 3-1

Overview 3-1

Display Instructions 3-1
tline 3-3
tcol 3-4
clear 3-5
out 3-6
rev 3-7
scroll 3-8
screen 3-9

Data Acquisition Instructions 3-10
Overview 3-10
decz 3-11
reqz 3-13
inz 3-14
outz 3-15
clearz 3-16
revz 3-17

Screen Page Management Instructions 3-18
Overview 3-18
ssave 3-19
sload 3-20

4 BASIC Complements..... 4-1

uplow 4-2
hunt 4-3
creatst 4-4

5 - DOS Support..... 5-1

Appendices
A - Summary of the Instructions
B - Error Messages
C - Product Comment Form.

VIC-MASTER, for the COMMODORE VIC personal computer is a powerful complement to your system, it facilitates, eases, improves greatly the performance of your programs.

It consists of a collection of new BASIC-type instructions that you will quickly learn to enhance the management of the screen, to draw lines and columns, to realize efficient data acquisition routines, to play with the colors and the screen.

VIC-MASTER also includes very useful instructions to utilize with string of characters, and also DOS support instructions to ease your work with the disk drive.

This document has been designed to be both a teaching aid and a reference manual. A careful reading is recommended to help you in understanding the utilization of VIC-MASTER. If you read the manual and try the examples on your VIC simultaneously, this should give you a fair idea of all the capacities of the product very rapidly.

OVERVIEW

VIC-MASTER is stored in a cartridge.

Just plug the cartridge in the memory expansion pinout located at the back of your VIC and power on the system. The full capacities of VIC-MASTER is now at your disposal.

Notes :

1) The hardware configuration of the cartridge should avoid any confusion, if in any doubts please check on your VIC's manual the connections of input/output devices.

2) You can also plug your VIC-MASTER cartridge in a Master Control Panel.

OVERVIEW

The screen generator integrated into MASTER was designed to ease data acquisition on the screen. The acquisition is more reliable than normal BASIC routines and the execution, fundamental in screen management, much quicker, since it is written in machine language.

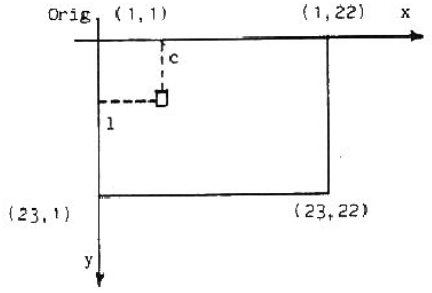
VIC-MASTER controls the following functions :

- display commands to draw lines, columns, to scroll on the screen, etc...,

- acquisition commands to define acquisition zones and associated controls, to allocate zone contents to BASIC variables,complete printusing, etc...,

- screen-page management commands includes save and load of screen-page from the disks.

In the following pages, the names we shall use for variables are arbitrary, they were chosen for easier comprehension. Any BASIC variables would suit, besides the reserved variable for VIC-MASTER "zo".



The screen is defined as a drawing board with the origin in the upper left-hand corner. The origin coordinates are 1,1.
Any point on the screen is defined by :

- l = line number
- c = column number

OVERVIEW (continued)

Generally speaking :

- n : line (or row) number (1 < = l < = 23)
- c : column number (1 < = c < = 22)
- ln : length (or number of columns)-horizontal line
- lg : height (or number of lines)-vertical line.

For a better understanding of the chapter, experiment these commands on the screen of your system, in direct or programmed mode.

A screen page is everything on a screen at a given time. It is possible to create complete pages with a frame or a board, save them on disk, load them directly on the screen, fill them with data, and a lot more.

tline

Function : tline draws on the screen a horizontal line defined by its origin and length.

Syntax : tline ln, l, c
with ln : length of the line,
l : line # of the starting point,
c : column # of the starting point.

Utilization : The tline command is used to draw frames on the screen,

Example : tline 22,1,1
tline 22,12,1

These two instructions draw two parallel lines in the upper part of the screen.

Notes :

To repeat the same instruction with different parameters several times, use the "←" symbol to avoid writing the body of the instruction each time.

The previous example becomes :

tline 22,1,1 ← 22,12,1

tcol

Function : tcol draws on the screen a vertical line defined by its origin and length.

Syntax : tcol ln, l, c
with ln : height of the vertical line
l : line # of the starting point
c : column # of the starting point.

Utilization : The tcol command is used to draw frames (with tline)

Example : tcol 12,1,1
tcol 12,1,22
These two instructions draw two parallel columns in the upper part of the screen.

Notes 1) Similarly to tline, ← symbol simplifies the coding
The previous example becomes :
tcol 12,1,1←12,1,22

2) 10 tline 22,1,1←22,12,1
20 tcol 12,1,1←12,1,22

These two BASIC lines draw a frame. Note that the intersections of the frame are correctly drawn.

clear

Function : clear clears the screen from an origin point.

Syntax : clear ln, l, c
with ln : length of the part to clear,
l : line # of the starting point,
c : column # of the starting point.

Utilization : clear is used to clear parts of the screen. The command clears only the parts of the display required other parts.

Examples : To clear the first line of the screen

clear 22,1,1

To clear the frame already drawn and also all that is inside the frame

30 for i = 1 to 12 : clear 22,i,1 : next i

(clears a "window" of 12 lines and 22 columns).

Notes : 1) A symbol can be used (see tline)
ex : clear 10,1,1←10,2,2

2) Before using the instruction scroll (see 3-8), and a print "♥", it is necessary to initialize the color screen of your VIC.

Type:
clear 255,1,1
clear 251,11,14

The screen will be automatically refreshed.

out

Function : out displays on the screen a string, a$, at a given location.

Syntax : out a$,l,c
with a$: any string
l : line # of the starting point
c : column # of the starting point.

Utilization : out displays a string on any part of the screen without having to use the cursor keys in string form. Out facilitates also frame filling.

Example
```
40 a$ = "JOHN SMITH"
50 out a$,5,10
```
These two lines will display JOHN SMITH on the fifth line starting at the tenth column.

Note : "←" can be used (see tline)

```
40 a$ = "JOHN SMITH" : b$ = "PATTY DAVIS"
50 out a$,5,10  b$,5,15
```


rev

Function : rev converts to reverse mode a window defined by its starting point, length, height and color.

Syntax : rev l,c,lg,ln [,co]

With :
l : line # of the origin point
c : column # of the origin point
ln: length (or number of columns)
lg: height (or number of lines)
co: new color : optional.

Utilization : rev draws the user's attention to a section of the

the same coordinates, it is possible to make the window blink. The speed of the blinking can be regulated by using timing loops
(for : next).

Example : To use the reverse mode for the names of the previous example.
```
140 rev 5,1,1,10,3←6,10,1,11,4
```

Note : You can use "←" (see tline)

scroll

Function : scroll shifts, up or down, left or right, part of the screen or window, defined by its starting point, its length and height.

Syntax :

```
scroll l,c,lg,ln,ty
with
l : line # of the starting point
c : column # of the starting point
ln: length of the window (or number of columns)
lg: height of the window (or number of line)
ty: type of scroll          - u for up
                            - d for down
                            - l for left
                            - r for right
```

Utilization : Scroll displays data files too large to be shown on one line. With simple coding, it is possible to scroll data in a frame and, as soon as the value is found, to save it and to carry on with the program.

Example :

```
10 print "  "
20 clear 255,1,1: clear 251,11,14
30 a$="JOHN":b$="PATTY":c$="JULIE"
40 out a$,11,4←b$,11,10←c$,11,17
50 for i=0 to 9
60 scroll 10-i,4,2,4,u
70 clear 4,11-i,4
80 scroll 11+i,17,2,5,d
90 clear 5,11+i,17
100 for j=1 to 200:next j
110 next i
120 end
```

Just type the program and see what happens. You can also run the program without the lines 70 and 90 to see the difference. You can do similar samples using the left and right scrolling.

Notes : ← Can be used (see tline)

- The height of a window is always greater than 1.
- A scroll does not erase the previous output.

screen

Function : screen modifies the colors of the screen, the borders and the characters displayed by VIC-MASTER.

Syntax : screen sc,br[,cr]
with sc : color of the screen (0<=sc<=15)
br : color of the border (0<=br<=7)
cr : color of the characters (0<=cr<=7)(optional)

Utilization :

Using the screen instruction, you can easely define the colors of your screen, border and characters displayed by VIC-MASTER (using tline, tcol, out, outz and reqz -see these instructions).

If the color mode is BLUE for the characters, when the VIC is turned on, the modification of this color using a SCREEN will be effective only for the characters displayed by VIC-MASTER instructions.

Examples :

```
clear 255,1,1:clear 251,11,14 : rem (see clear 3-5)
? "♥ ":screen 1,6,2:out"U.S.A",11,9
? "♥ ":screen 1,5,2:out"ITALY",11,8
? "♥ ":screen 0,2,7:out"GERMANY",11,7
? "♥ ":screen 1,2,2:out"JAPAN",11,9
```

Note : 1) Color code table

Parameters	Color	Code
For	Black	0
	White	1
SC	Red	2
BR	Cyan	4
	Purple	5
and	Green	5
CR	Blue	6
	Yellow	7
	Orange	8
	Light Orange	9
For	Rose	10
	Light Cyan	11
SC	Light Magenta	12
	Light Green	13
only	Light Blue	14
	Light Yellow	15

2) STOP and RESTORE will provide the original colors white and blue.

Overview :

to be used instead of INPUT and GET to allow homogeneous, controlled, efficient and reliable data acquisition.

This section is dedicated to MASTER acquisition commands. A zone is defined by certain parameters (starting point, size, allowed type of characters, possibilities of exit (other than return) and an identification number). (decz)

Once the zone is defined, instruct MASTER to bring data in it, perform the controls assigned and reproduce the screen control functions, such as : moving the cursor, deleting, inserting (reqz) . The inz command performs information transfer from a zone to a string (outz performing the opposite operation). 128 different zones can be defined on one page or screen.

```
input a$ <-------> [decz 1   : zone definition
                   reqz 1    : controlled acquisition
                   inz 1,a$: transfer into a$
```

We will give more details in the following pages.

decz

Function : decz is used to define all parameters of a screen zone for

Syntax : decz n,l,c,ln [,ty] [,f$]
with n : identification number (0<=n <=127)
l : line # of the starting point (1<= l <=23)
c : column # of the starting point (1<= c <=22)
ln : total length of the zone (1<= ln <=255)
ty : type of control zone
value of ty :
. ty = n numeric data only,
. ty = m data in capital letters,
. ty = r other possibilities of exit than with return,
. for a numeric zone : all the keys, numeric ones excepted,
. for an alphanumeric zone : ⇑ , ⇓ , CTRL + 0, CTRL+9
. The ASCII code of the key you press for exit is put in za, so that you can define control keys,
⇑ : za=145
⇓ : za=17
CTRL+9:za=18
CRTL+0:za=146
. ty = p zone defined in printusing + f$

. f$: string containing the format for the print using.

A format can define a zone. When bringing or displaying data in this zone, its contents is formatted. The format is stored in a string (here : f$). The control characters are :

9 : numeric,
8 : numeric; set to zero if nil
1 : sign position (only -)
2 : sign position (with + or -)

Any alphanumeric character can be inserted into a format.
Ex : f$ = "$ 19,999,998.88"

Utilization : decz allows to define, as precise as possible, the aquisition zone. It is possible to work efficiently and easely on a zone just by using its identification number. The frames or boards management is simplified.

decz

Example : decz 2,1,2,10,n,p,"199 999.88"

This exemple defines a zone (number 2) with a starting point (1,2) of 10 characters. Only numbers will be accepted (n). Printusing format is used with the printing of a sign only if it is a minus and zero being forced at the end of the number. The number will be 5 digits long with 2 digits after the decimal point.

Notes : . ← can be used (see tline)
. more than one type can be combined in the same decz.
all these different possibilities are excepted.

n	: Only numbers	-RETURN to leave the zone
n,r	: Only numbers	-A programmable key to leave the zone
n,p	: Only numbers using a format	-RETURN to leave the zone
n,r,p	: Only numbers using a format	-A programmable key to leave the zone
m	: Only capital letters	-RETURN to leave the zone
m,r	: Only capital letters	-A programmable key to leave the zone
r	: Every character accepted	-A programmable key to leave the zone
p	: (equivalent to n,p)	
r,p	: (equivalent to n,r,p)	

(nothing): every character is accepted. RETURN to leave the zone.

If the data overflows the aquisition zone, the zone is highlighted, and the whole zone erased. The cursor moves back to the beginning of the zone . This prevents leaving the zone if the format is not correct (the format is controlled when you press RETURN).

If you try decz instructions without using a reqz after it, nothing will happen but the declaration will remain.

reqz

Function : reqz performs the acquisition using the parameters defined in the corresponding decz.

Syntax : reqz n
With n : zone identification number

Utilization : reqz is used in association with decz. If a corresponding decz doesn't exist (or hasn't been loaded in a screen-page), an error message will be generated.

With this command, MASTER takes the data acquisition under its control, as defined in the zone declaration. On execution of this command results a cursor displayed at the beginning of the zone.

The CLR, HOME, CURSOR BACK and FORWARD, INST/DEL functions can be used in the zone.

The data acquisition is controlled as defined in the zone declaration.

If the mode to define the zone is r, and the exit code different from 13 (zo), the cursor will return to the position occupied on exit from zone if this zone is re-entered.

Example : If the previous example has been entered (decz 2,...), juste type reqz 2 to see what happens. Only the numbers will be accepted.

Notes : "←" is allowed (see tline), but if more than one decz of r type (using a programmable key to leave the zone), is used, something like reqz 1←2←3 cannot be utilized because zo is not tested after each reqz.

inz

Function : inz transfers the contents of a zone to a string variable.

Syntax : inz n,a$

a$: any string variable.

Utilization : inz is used to transfer zone contents to a string when all data in a zone are satisfying.

Example : inz 2,a$
If the two previous examples have been tried, this one should have transfered the contents of the zone into a$. To verify, just type
out a$,3,5

Notes . ← is possible (see tline)
. The string variable receiving the contents of a zone will have the same length as the zone, whatever the contents of the zone is.

outz

Function : outz displays a string in a zone.

Syntax : outz n,a$

a$: any string variable.

Utilization : outz is the opposite instruction of inz. Data goes from a variable to a zone. This instruction is similar to the display instruction out but it is easier to use and more precise.

Easier because the zone number instead of the coordinates of the starting point is required.

More precise because it assumes all the controls defined in the zone by the decz.

Example : still using the previous examples
a$ = "11111"
outz 2,a$

Notes : . "←" still available (see tline)

. in case of a zone defined in printusing
if an overflow of decimals occurs, the number is truncated
If an overflow of capacity occurs, the zone is filled with"*****".

. in case of an outz in a zone declared with no sign (print using without 1 or 2), of a string with a +,- or blank as a first character, the string can be shortened by a digit, or the display can be replaced by stars. To avoid that, you have to extract the sign by doing:

a = len(mo$) - 1 : mo $ = right$(mo$,a)

clearz

Function : clearz clears a zone.

Syntax : clearz n1 [,n2 to n3] [,...]
with n1,n2,n3 zone identification numbers.

Utilization : clearz can be used to "clear" the contents of a frame, without clearing the frame itself, after an acquisition and before a new acquisition.

Example : clearz 2

revz

Function : revz converts to reverse mode a zone.

Syntax : revz n1 [,n2 to n3][,...]

Utilization : revz can be used to bring the user's attention to the contents of a zone, for a message or a result. It is possible to blink the zone (see rev 3-7).

Example : revz 2

OVERVIEW

In a business application, the data acquisition frames are often similar and are used more than once with different information. In a normal BASIC program, a new frame has to be created each time, wasting both time and memory.

MASTER defines a screen page, with all the decz (declarations) instructions, and saves the page on disk, avoiding the necessity of creating a frame over and over again. Different applications can have their own screen page. Coding is simplified.

The page can be loaded on the screen later and filled with data. It is possible to exchange an actual screen page with a page in memory, for exemple, a HELP screen.

If during an acquisition, the control commands are forgotten, the program can easily exchange pages for a HELP page, read the commands, and go back to the previous pages. The cursor will reappear at the exact point it was before the pages exchange.

The two instructions perform the following action :

- saving a page ssave 3-19
- loading a page sload 3-20

ssave

Function : ssave saves a screen page on disks.

Syntax : ssave du, "name" [,n1] [,n2 to n3] [,..]
with
du : disk unit logical number (usually 8)
n1,n2,n3 : associated zones

Utilization : The current screen page will be saved. The zones (decz) can also be saved, using the syntax, n1 or , n2 to n3. This syntax is optional. In all cases, the entire page will be saved (frames, boards, data ou display...)

Example : ssave 8,"prg1",1,12,30 to 70

Note : ssave 8,"name",0 to 127 would save all associated declarations. There are no errors if non-declared zones are saved.

sload

Function : sload loads on the screen a page saved on disk.

Syntax : sload du,"name"
with
du : disk unit logical number (usually 8)

Utilization : Same principle as ssave.

The page will be directly displayed on the screen. All the zones will be declared again.

Example :
sload 8,"prog1"

Note : none

Overview :

Grouped under this chapter are three functions which do not depend directly on the screen generator. These instructions simplify the controls, improve some performances of the BASIC and ease the programming, specially for the use of strings of characters.

These functions include :

- upper/lower case conversion 4-2
- locating characters inside a string 4-3
- creation of a string of characters 4-4

These functions are all independant and can be studied separately.

uplow

Function : uplow convert the case of characters of a string.

Syntax : uplow a$
with a$: any string

Utilization : In a string, all upper case characters are changed to lower case and vice-versa.

Example : a$="PATRICIA"
uplow a$
out a$,5,5
patricia

Note : None

hunt

Function : hunt will find the position of a character within a string.

Syntax : hunt [-] chr$(ca),a$,pn
with - : optional symbol
with - : search for the first character equal to
without-search for the first character not equal to
ca : decimal value of an ASCII character
a$: string where the search is to be done
pn : pointer within the string to indicate where the search starts.

Utilization :

hunt controls opérations such as acquisition, validation and also statistics. The result will be found in zo.

Example : a$ = "patricia owen"
hunt chr$(65),a$,1:rem 65 : a
? zo
2
hunt-chr$(69),a$,1:rem 69 : e
?zo
1

Note : If the character does not exist, zo=0

Creation of a string

creatst

Function : creatst creates a string

Syntax : creatst a$, ln [,chr$ (ca)]
with ca : value of an ASCII character
ln : wanted length of the string
a$: name of the string

Utilization :

The creatst command creates a string of ln characters long (chr$(ca)). In the absence of chr$ (ca), chr$ (32) will be used. You can use any character you want.

Example : creatst a$, 80," "
will create a string a$ of 80 blanks

```
creatst b$,10,"*"
```

will create a string b$ with 10 stars.

Note :

This instruction replaces the following BASIC loop

```
for i = 1 to ln
a$ = a$ + chr$(ca)
next i
```


Overview

VIC-MASTER simplifies the operations for all disk's user in including the DOS support which will allow you to use quick commands to manage the loading, saving, etc ... on diskette.

@ : will give the disk error message,

@$: will give the catalog of your diskette,

@I : will initialize the disk unit,

@V : will perform a collect,

@N : filename, identifier
: will format a diskette,

@R : new filename = old filename
: will rename a file or program,

@S : filename
: will scratch a file or program.

@C : new filename, to old filename, new drive
: will copy a file from a drive to another or make a copy on the same one.

All commands are to be followed by hitting the RETURN key.

Note : For more details of these instructions and their syntax you can refer to your VIC-1540 User's manual.

Instructions

clear ln,l,c	- erases ln characters (starting point :l,c)	3-5
clearz n	- erases the nth zone.	3-16
creatst a$,ln[,chr$(ca)]	- creates a string called a$	4-1
decz n,l,c,ln[,ty][,f$]	- declares the nth zone.	3-11
hunt[-]chr$(ca),a$,pn	- searchs for a character in a string	4-2
inz n,a$	- transfers the content of the nth zone into a$	3-14
out a$,l,c	- displays a$ (origin point :l,c)	3-6
outz n,a$	- displays a$ into the nth zone and contols a$	3-15
reqz n	- requires an answer in the nth zone	3-13
rev l,c,lg,ln[,co]	- converts to reverse mode on part of the screen	3-7
revz n	- converts to reverse mode into the nth zone	3-17
screen sc,br[,cr]	- modifies the colors of the screen, the borders and the characters	3-9
scroll l,c,ln,lg,ty	- scrolls a part of the screen	3-8
sload 8,"0:name"	- loads a screen page in memory (screen)	3-20
ssave 8,"0:name"[,n][,n2ton3]	- saves a screen page on disk with the zones	3-19
tcol ln,l,c	- draws a line of length ln (origin l,c)	3-4
tline ln,l,c	- draws a col. of length ln (origin l,c)	3-3
uplow a$	- upper case/lower case convertion	4-2

Reserved variables

zo	- ASCII code for the return key in request mode (decz)
	- pointer giving the location of a character found with a hunt.

Not-Reserved Variables (Programmer's Choice)

a$	- string - transfer
br	- color of the border of the screen
c	- column number (x-coordinates)
ca	- ASCII value of a character
co	- color (for the rev instruction)
cr	- color of the characters
f$	- print-using format
l	- line number (y-coordinates)
lg	- width(or height) of a window(zone)(-number of lines).
ln	- length of a window (zone) - number of columns
	- length of a string (creatst)
n1,n2,n3,n	- zone identification number
pn	- pointer within a string (hunt)
sc	- color of the screen
ty	- formatting type (decz)
	- scrolling type (scroll)-(u,d,l,r)

List of the error messages generated by VIC-MASTER with a comment of the problem.

BAD FORMAT

decz using the p-type :
the print-using defined does not correspond to the variables used - length problems (see 3-11)

NO COLOR

A color parameter is missing or the value of a parameter is not allowed. (see instruction 3-9)

NOT IMPLEMENTED

Equivalent to a syntax error. You have used a token used by VIC-MASTER in higher versions.

NO ZONE

A clearz, outz or revz has been performed on a non-declared zone (see 3-15,3-16,3-17)

OUT OF PAGE

coordinates of the starting point or length or height are incorrect in the screen generator instructions.

VIC-MASTER

PRODUCT COMMENTS FORM

Now that you know the product qualities and weaknesses, we would very much appreciate if you could fill this form . Your comments will assist us in improving our product to serve you better.

If you wish a reply from us or/and to be on MICRO APPLICATION mailing list, write your name and address below.

Name :
Address :
City : Country (state) :
Postcode :

VIC-MASTER

Which are the most useful functions ?

Which are the less useful functions ?

Do you use the utilities ? (Which ones?)

Which improvements would you like to see ?

Which kind of applications do you use MASTER for ?

Bugs (mistakes) ?

Notes :

DOCUMENTATION

How did you find this manual ?

Strong-points ?

Weak points ?

Notes ?

FUTURE PRODUCTS

What kind of products (hardware or sofware) would you like to see on your computer ?

Notes :

Thank you

Please send this form to : MICRO APPLICATION
73 rue de Turbigo 75003 PARIS FRANCE.